EECS 151/251A Homework 3

Due Monday, Feb 15th, 2021

Please include a short (1-2 sentence) explanation with each answer unless otherwise directed in the question.

Problem 1: State Elements

Consider a 3-bit Linear Feedback Shift Register (LFSR). This circuit is made up of 3 positive edge-triggered flip-flops in a delay chain. the input to `DFF0` is the XOR of the outputs of `DFF1` and `DFF2`. The inputs are therefore described as follows.

$$d_0 = q_1 \oplus q_2$$
$$d_1 = q_0$$
$$d_2 = q_1$$

Draw the circuit diagram of the LFSR. Provide a timing diagram (i.e. waveform) of all the register outputs (q_0, q_1, q_2) and the output of the XOR (d_0) for 10 cycles of the LFSR with the registers initialized to `q[2:0]`=001. Use a clock period of 1 ns. The registers and XOR gates are *realistic* and have delays ($\tau_{\text{clk-q}} = 100\,\text{ps}$, $\tau_{\text{setup}} = 10\,\text{ps}$, $\tau_{\text{hold}} = 0\,\text{ps}$ for the register, and $\tau_{p,\text{XOR}} = 50\,\text{ps}$).

Problem 2: FPGA Logic Cell

You've started your new job at an FPGA manufacturing startup and after getting settled in you are handed your first big assignment: the company is pursuing a different kind of logic cell and have asked you to be the design lead for it! This cell will function as either a 5-LUT, or as two *independent* 4-LUTs (that is the block can support up to 8 data inputs and 2 outputs), with registered outputs. The foundry has given you a library of LUTs (of any size), multiplexers (up to 8-to-1) and flip-flops, but for some reason has decided not to provide any other logic elements. It's been a while since the last financing cycle for your company and funds are pretty tight, so you are asked to minimize the amount of hardware required while still achieving the same behavior. Provide the circuit diagram for your proposal as well as a 4-5 sentence abstract of how your circuit works and the design decisions made.

Problem 3: Bit-stream Reverse Engineering

You have been recruited as a penetration tester for a company manufacturing FPGA-based secure endpoints for a private network. They have asked you to pose as a potential attacker and try to find vulnerabilities in their system. After thinking about the most likely avenues of attack, you decide to pose as a malicious actor who has acquired detailed information on the FPGA that is being used in the endpoints. After analyzing a sample of the device, you were able to determine a few properties of the system.

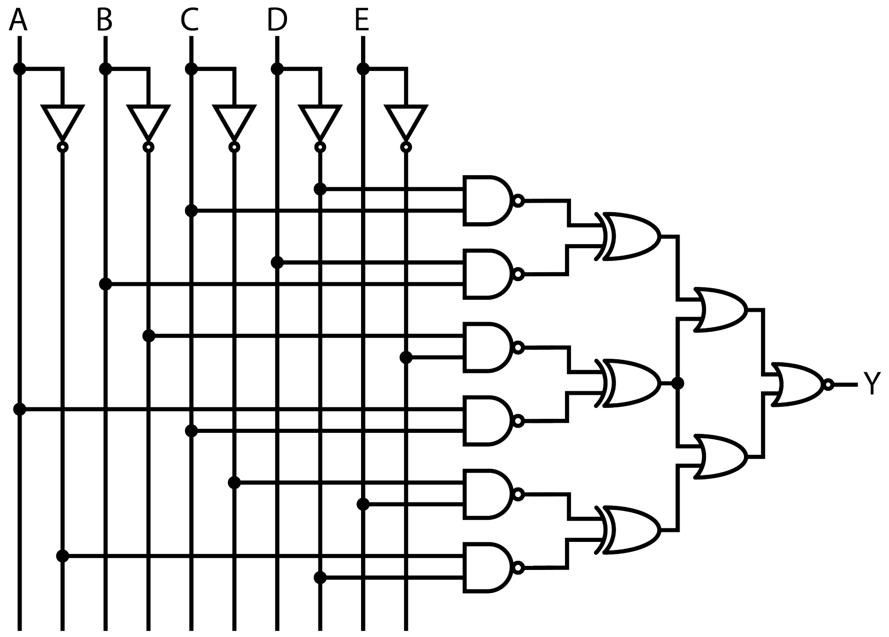
1. The device contains a collection of N-LUTs (the value of N is part of the mystery). The LUTs are numbered 0, 1, 2, ... Each LUT in the FPGA has the same number of inputs (same N).
2. Each LUT has an output labeled y_i, where i is the LUT number, and inputs labeled x_{i_j}, where i is the LUT number and j is the input number.
3. For programming, the LUTs are connected in a shift register. They are programmed with a configuration bit-stream shifted in from one LUT to the next. Recall from lecture that the configuration bit-stream programs the values of the latches for the truth table.
4. The encryption is a stream cipher. This particular implementation uses several LFSRs, **one of which is relying on some LUTs to perform an XOR operation**. If the XOR LUTs are exposed, it would seriously compromise the security of the device, so the company is very interested in knowing which LUTs are exposed.
5. One of the LUTs is programmed as a simple 2-bit AND gate (*Hint: this will be useful in figuring out the endianness and size of the LUTs.*).
6. Via a side-channel power analysis attack, you were able to determine part of the bitstream, shown here: `0x1111111169969669575757FF699696697F77FFFF`. This bitstream is fed in from right to left (i.e. `F` is fed first and `1` is fed last).

(a) How many LUTs would the above bit stream program?

(b) For each of the LUTs, determine the boolean function performed. Show your work.

(c) You performed the side-channel attack only after realizing the FPGA was beginning to be reprogrammed and so only got the tail end of the programming bitstream (that is the last LUT you see being programmed is `LUT0`). Which LUTs should you report are the XOR LUTs that are vulnerable to detection?

Problem 4: LUT Mapping



(a) Using only 4-LUTs, partition the circuit above into as few LUTs as possible. *Do not simplify the gate-level circuit before mapping to LUTs.*

(b) Using only 3-LUTs, partition the circuit above into as few LUTs as possible. *Do not simplify the gate-level circuit before mapping to LUTs.*

Problem 5: State-of-the-Art FPGAs

Time to do a little research! Find the latest and largest FPGA from Altera (now Intel). How many logic cells are on it? How many LUTs are on each logic cell? How many inputs per LUT?

Problem 6: K-Maps

Consider the following SOP expression:

$$\begin{aligned} y = {} & a'b'c'd'e' + a'b'c'd'e + a'b'c'de' + a'b'c'de + a'b'cd'e + a'b'cde \\ & + a'b'cde' + a'bcd'e' + a'bcd'e + a'bcde + a'bcde' \\ & + abc'd'e' + abc'de' + ab'cd'e' + ab'cd'e + ab'cde \\ & + ab'cde' + abcd'e' + abcd'e + abcde + abc'd'e \\ & + abc'de \end{aligned}$$

Provide the truth table for this expression. Simplify this expression with the help of a 5-variable K-map into the minimal SOP form.

Problem 7: Another K-Map

As the head of circuit design at a small IoT device startup, you have been tasked with designing the interface between one of the environmental sensors and the microprocessor. The analog designer has created a front-end interface that reads in the voltage from a barometer to a simple resistor ladder ADC, which converts the sensor input into a custom *thermometer* code. You are now tasked with decoding this thermometer code to binary so the processor can read the value. The truth table is below.

T_3	T_2	T_1	T_0	b_1	b_0
0	0	0	1	0	0
0	0	1	1	0	1
0	1	1	1	1	0
1	1	1	1	1	1

Note that in this case binary code 00 corresponds to a thermometer code of 1

What is the minimal SOP form for each binary bit as a function of the thermometer bits? Use a K-map to help your reduce the terms.

Problem 8: Boolean Algebra

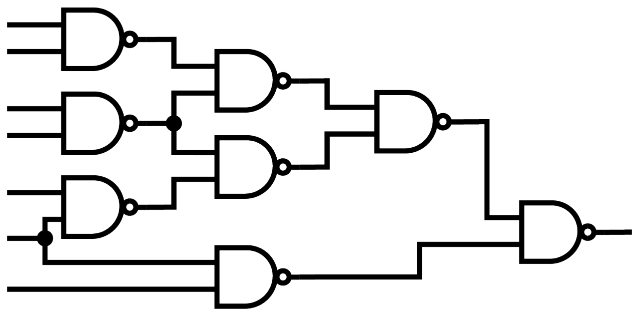
Consider a full subtractor, the negative counterpart of the full adder. This block has two outputs, d and b_{out}, for the difference and borrow outputs, respectively.

(a) Provide the truth table for this block, with the inputs x, y, and b_{in}.

(b) Perform a simplification by boolean algebra of the b_{out} output. Refer to slide 25 of Lecture 6 for an example of boolean simplification.

Problem 9: De Morgan's Law

(a) Convert the following circuit into an equivalent circuit using **only** a minimal number of 2-input NOR gates.



(b) The d output for the full subtractor in Problem 7 can be expressed in SOP form as

$$d = x'y'b_{in} + xy'b'_{in} + x'yb'_{in} + xyb_{in}$$

Convert this to POS form.